CLAIMS

1. A network interface adapter, comprising:

a host interface, for coupling to a host processor;

an outgoing packet generator, adapted to generate an outgoing request packet for delivery to a remote responder responsive to a request submitted by the host processor via the host interface;

a network output port, coupled to receive the request packet from the output packet generator, so as to transmit the outgoing request packet over a network to the remote responder;

a network input port, for coupling to the network so as to receive an incoming response packet from the remote responder, in response to the outgoing request packet sent thereto, and further to receive an incoming request packet sent by a remote requester;

an incoming packet processor, coupled to the network input port so as to receive and process both the incoming response packet and the incoming request packet, and further coupled to cause the outgoing packet generator, responsive to the incoming request packet, to generate, in addition to the outgoing request packet, an outgoing response packet for transmission via the network output port to the remote requester.

2. An adapter according to claim 1, wherein the outgoing request packet comprises an outgoing write request packet containing write data taken from a system memory accessible via the host interface, and

wherein the outgoing response packet comprises an outgoing read response packet containing read data taken

from the system memory in response to the incoming request packet, and

wherein the outgoing packet generator comprises a gather engine, which is coupled to gather both the write data and the read data from the system memory for inclusion in the respective outgoing packets.

3. An adapter according to claim 2, wherein to submit the request, the host processor writes a request descriptor indicative of the write data to a first memory location, and wherein to cause the outgoing packet generator to generate the outgoing response packet, the incoming packet processor writes a response descriptor indicative of the read data to a second memory location, and wherein the gather engine is adapted to read information from the descriptors and to gather the read data and the write data responsive thereto.

4. An adapter according to claim 1, wherein the outgoing packet generator comprises a plurality of schedule queues, and is adapted to generate the outgoing request packet and the outgoing response packet responsive to respective entries placed in the queues.

5. An adapter according to claim 4, wherein the network input and output ports are adapted to receive and send the incoming and outgoing packets, respectively, over a plurality of transport service instances, and

wherein the outgoing request packet and the outgoing response packet are associated with respective instances among the plurality of transport service instances, and

wherein the outgoing packet generator is adapted to assign the transport service instances to the queues based on service parameters of the instances, and to

place the entries in the schedule queues corresponding to the transport service instances with which the incoming and outgoing packets are associated.

6. An adapter according to claim 5, wherein the outgoing packet generator comprises:

one or more execution engines, which are adapted to generate the outgoing request packet and the outgoing response packet responsive to a list of work items respectively associated with each of the transport service instances; and

a scheduler, which is coupled to select the entries from the queues and to assign the instances to the execution engines for execution of the work items responsive to the service parameters.

7. An adapter according to claim 5, wherein the transport service instances comprise queue pairs.

8. An adapter according to claim 4, wherein the outgoing packet generator comprises one or more doorbell registers, to which the host processor and the incoming packet processor write in order to place the entries in the queues.

9. An adapter according to claim 4, wherein the incoming request packet comprises a write request packet carried over the network on a reliable transport service, and wherein responsive to the incoming write request packet, the incoming packet processor is adapted to add an entry to the entries placed in the queues, such that responsive to the entry, the outgoing packet generator generates an acknowledgment packet.

10. An adapter according to claim 1, wherein the incoming request packet comprises an incoming read request packet, and wherein responsive to the incoming read request packet, the incoming packet processor is adapted to prepare a read response work item in a memory location, and wherein the outgoing packet generator is coupled to read the read response work item from the memory location and, responsive thereto, to generate a read response packet.

11. An adapter according to claim 10, wherein the incoming packet processor is configured so that when it receives an incoming write request packet containing write data to be written to a system memory accessible via the host interface after receiving the incoming read request packet, it conveys the write data to the host interface without waiting for execution of the read response work item.

12. An adapter according to claim 10, wherein the incoming packet processor is configured so that when it receives an incoming write request packet containing write data to be written to a system memory accessible via the host interface before receiving the incoming read request packet, it prevents execution of the read response work item until the write data have been written to the system memory.

13. An adapter according to claim 1, wherein the incoming response packet comprises an incoming read response packet sent by the remote responder in response to the outgoing request packet, the incoming read response packet containing read data to be written to a system memory accessible via the host interface, and

wherein the incoming request packet comprises an incoming write request packet containing write data to be written to the system memory, and

wherein the incoming packet processor comprises a scatter engine, which is coupled to scatter both the write data and the read data from the respective incoming patents to the system memory.

14. An adapter according to claim 1, wherein the outgoing packet generator is adapted, upon generating the outgoing request packet, to notify the incoming packet processor to await the incoming response packet so as to write a completion message to the host interface when the awaited packet is received.

15. An adapter according to claim 1, wherein the incoming request packet comprises an incoming read request packet specifying data to be read from a system memory accessible via the host interface, and

wherein the incoming packet processor is adapted to write a response descriptor to a memory location indicating the data to be read from the system memory responsive to the read request packet, and

wherein the outgoing packet processor is adapted to read the response descriptor from the memory location and, responsive thereto, to read the indicated data and to generate the outgoing response packet containing the indicated data.

16. An adapter according to claim 15, wherein the incoming read request packet is one of a plurality of incoming read request packets, and wherein the incoming packet processor is adapted to write the response descriptor to the memory location as part of a list of

such descriptors, responsive to which the outgoing packet processor is adapted to generate the outgoing response packet as part of a sequence of such packets.

17. An adapter according to claim 16, wherein the network input and output ports are adapted to receive and send the incoming and outgoing packets, respectively, over a plurality of transport service instances, and wherein the incoming packet processor is adapted to prepare the list of the response descriptors for each of the instances as a part of a response database held for the plurality of the instances in common.

18. An adapter according to claim 17, wherein the transport service instances comprise queue pairs.

19. An adapter according to claim 15, wherein the request comprises a write request, which is submitted by the host processor by generating a request descriptor indicating further data to be read from the system memory for inclusion in the outgoing request packet, and wherein the output packet generator is adapted to read the request descriptor and, responsive thereto, to generate the outgoing request packet as a write request packet containing the indicated further data.

20. A network interface adapter, which comprises a plurality of circuit elements arranged on a single integrated circuit chip, the elements comprising:

a host interface, for coupling to a host processor and to host system resources associated with the host processor;

a network input port, for coupling to a network so as to receive incoming read request packets sent by a

remote requester, specifying data to be read via the host interface;

an incoming packet processor, coupled to the network input port so as to receive and process the incoming read request packets, and further coupled to a memory off the chip so as to write a list of descriptors to the memory indicating the data to be read in response to the incoming read request packets;

an outgoing packet processor, coupled to the host interface so as to read the list of descriptors from the memory and, responsive thereto, to read the indicated data and to generate outgoing response packets containing the indicated data; and

a network output port, coupled to receive the outgoing response packets from the outgoing packet processor so as to transmit the outgoing response packets over the network to the remote requester.

21. An adapter according to claim 20, wherein the outgoing packet processor comprises a doorbell register, and wherein the incoming packet processor is coupled to write to the doorbell register in order to signal the outgoing packet processor to read the list.

22. An adapter according to claim 20, wherein the network input and output ports are adapted to receive and send the incoming and outgoing packets, respectively, over a plurality of transport service instances, and wherein the incoming packet processor is adapted to write the descriptors to a plurality of lists corresponding to the plurality of the transport service instances.

23. An adapter according to claim 22, wherein the incoming packet processor is adapted to maintain the

plurality of the lists in a response database held in the memory for all the instances in common.

24. An adapter according to claim 23, wherein each of the instances is assigned a respective number of entries in the database to which its descriptors can be written.

25. An adapter according to claim 24, wherein the entries for each of the instances are arranged in the database in a cyclic buffer.

26. An adapter according to claim 22, wherein the transport service instances comprise queue pairs.

27. An adapter according to claim 22, wherein the outgoing packet generator comprises a plurality of schedule queues and is adapted to generate the outgoing response packets responsive to entries placed in the queues, each of the entries corresponding to one of the transport service instances for which the lists were prepared by the incoming packet processor.

28. An adapter according to claim 27, wherein the transport service instances are assigned to the queues based on service parameters of the instances, and wherein the outgoing packet generator comprises a scheduler, which is coupled to select the entries from the queues for service responsive to the service parameters.

29. An adapter according to claim 22, wherein each of the descriptors occupies a given volume of space in the off-chip memory, and wherein a maximum number of incoming read requests, generated responsive to the incoming read request packets, that can be outstanding at any given time is determined by the space available in the off-chip memory.

30. An adapter according to claim 20, wherein the system resources associated with the host processor comprise a system memory, and wherein at least a portion of the off-chip memory to which the list of descriptors is written is comprised in the system memory.

31. A method for coupling a host processor to a network, comprising:

generating an outgoing request packet for delivery to a remote responder using an outgoing packet generator, responsive to a request submitted by the host processor;

transmitting the outgoing request packet from the output packet generator over the network to the remote responder;

receiving an incoming response packet from the remote responder, in response to the outgoing request packet sent thereto, using an incoming packet processor;

receiving an incoming request packet sent by a remote requester using the incoming packet processor; and

coupling the incoming packet processor to the outgoing packet generator so as to cause the outgoing packet generator to generate, responsive to the incoming request packet, in addition to the outgoing request packet, an outgoing response packet for transmission via the network to the remote requester.

32. A method according to claim 31, wherein generating the outgoing request packet comprises generating an outgoing write request packet containing write data taken from a system memory associated with the host processor, and

wherein coupling the incoming packet processor to the outgoing packet generator comprises generating, using

the outgoing packet generator, an outgoing read response packet containing read data taken from the system memory in response to the incoming request packet, and

wherein generating the outgoing write request packet and generating the outgoing read response packet comprise generating the packets using a gather engine in the outgoing packet generator, which is coupled to gather both the write data and the read data from the system memory for inclusion in the respective outgoing packets.

33. A method according to claim 32, wherein generating the outgoing write request packet comprises generating a request descriptor indicative of the write data to a first memory location, and wherein generating the outgoing read response packet comprises writing, using the incoming packet processor, a response descriptor indicative of the read data to a second memory location, and wherein generating the packets using the gather engine comprises reading information from the descriptors using the gather engine and gathering the read data and the write data responsive thereto.

34. A method according to claim 32, wherein the outgoing packet generator comprises a plurality of schedule queues, and wherein generating the packets comprises generating the outgoing request packet and the outgoing response packet responsive to respective entries placed in the queues.

35. A method according to claim 34, wherein the outgoing request packet and the outgoing response packet are associated with respective instances among a plurality of transport service instances in use on the network, and wherein generating the outgoing request packet and the

outgoing response packet comprises assigning the transport service instances to the queues based on respective service parameters of the instances, and placing the entries in the queues corresponding to the instances with which the packets are associated.

36. A method according to claim 35, wherein generating the outgoing request packet and the outgoing response packet comprises allocating resources to process the queues responsive to the respective service parameters.

37. A method according to claim 35, wherein the transport service instances comprise queue pairs.

38. A method according to claim 34, wherein receiving the incoming request packet further comprises receiving an incoming write request packet on a reliable transport service, and wherein generating the outgoing response packet comprises adding an entry to the entries in the queues, causing the outgoing packet generator, responsive to the entry, to generate an acknowledgment packet.

39. A method according to claim 34, wherein generating the outgoing write request packet and generating the outgoing read response packet both comprise writing to doorbell registers of the outgoing packet generator in order to place the entries in the schedule queues.

40. A method according to claim 31, wherein receiving the incoming request packet comprises receiving an incoming read request packet, and wherein coupling the incoming packet processor to the outgoing packet generator comprises preparing a read response work item using the incoming packet processor, and executing the

read response work item, using the outgoing packet generator, to generate a read response packet.

41. A method according to claim 40, wherein receiving the incoming request packet further comprises receiving an incoming write request packet containing write data to be written to a system memory associated with the host processor after receiving the incoming read request packet, and comprising conveying the write data to the system memory using the incoming packet processor without waiting for execution of the work item associated with the outgoing read response.

42. A method according to claim 40, wherein receiving the incoming request packet further comprises receiving an incoming write request packet containing write data to be written to a system memory associated with the host processor before receiving the incoming read request packet, and comprising conveying the write data to the system memory using the incoming packet processor while preventing execution of the work item associated with the outgoing read response until the write data have been written to the system memory.

43. A method according to claim 31, wherein receiving the incoming response packet comprises receiving an incoming read response packet sent by the remote responder in response to the outgoing request packet, the incoming read response packet containing read data to be written to a system memory associated with the host processor, and wherein receiving the incoming request packet comprises receiving an incoming write request packet containing write data to be written to the system memory, and comprising scattering both the write data and

the read data from the respective incoming patents to the system memory using a scatter engine in the incoming packet processor.

44. A method according to claim 31, wherein transmitting the outgoing request packet comprises passing a notification from the output packet generator to the incoming packet processor to await the incoming response packet to be received in response to the outgoing request packet, and comprising writing a completion message to the host processor when the incoming packet processor receives the awaited packet.

45. A method according to claim 31, wherein receiving the incoming request packet comprises receiving an incoming read request packet specifying data to be read from a system memory associated with the host processor, and

wherein coupling the incoming packet processor comprises writing a response descriptor to a memory location indicating the data to be read therefrom responsive to the read request packet, and causing the outgoing packet processor to read the response descriptor from the memory location and, responsive thereto, to read the indicated data from the system memory and to generate the outgoing response packet containing the indicated data.

46. A method according to claim 45, wherein receiving the incoming read request packet comprises receiving a plurality of incoming read request packets, and wherein writing the response descriptor comprises writing a list of such descriptors to the memory location, causing the

outgoing packet processor to generate the outgoing response packet as part of a sequence of such packets.

47. A method according to claim 46, wherein receiving the plurality of incoming read request packets comprises receiving the packets over a plurality of transport service instances on the network, and wherein writing the list of the descriptors comprises writing a respective list for each of the plurality of the instances to a response database held for the plurality of the instances in common, causing the outgoing packet processor to generate the packets for transmission over the plurality of the instances.

48. A method according to claim 47, wherein the transport service instances comprise queue pairs.

49. A method according to claim 45, wherein the request comprises a write request, which is submitted by the host processor by generating a request descriptor in a memory location indicating further data to be read from the system memory for inclusion in the outgoing request packet, and wherein generating the outgoing request packet comprises reading the request descriptor from the memory location and, responsive thereto, generating a write request packet containing the indicated further data.

50. A method for coupling a host processor and a system memory associated therewith to a network, comprising:

receiving at a network interface adapter chip coupled to the host processor incoming read request packets sent by remote requesters over respective transport service instances on the network, the read

request packets specifying data to be read from the system memory;

writing descriptors using the network adapter chip, responsive to the incoming read request packets, in a plurality of lists in an off-chip memory, the lists corresponding respectively to the transport service instances, the descriptors indicating the data to be read from the system memory;

reading the lists of descriptors from the off-chip memory and, responsive thereto, reading the indicated data and generating outgoing response packets containing the indicated data; and

transmitting the outgoing response packets to the remote requesters over respective transport service instances on the network.

51. A method according to claim 50, wherein the transport service instances comprise queue pairs.

52. A method according to claim 50, wherein reading the lists of the descriptors comprises writing to a doorbell register of the network interface adapter chip in order to signal the network interface adapter chip to read the lists and to generate the outgoing response packets responsive thereto.

53. A method according to claim 52, and comprising assigning the transport service instances to respective schedule queues, and placing entries in the schedule queues after writing the descriptors to the off-chip memory, each of the entries corresponding to one of the transport service instances having one of the lists corresponding thereto, wherein reading the lists of descriptors comprises selecting the entries from the

queues and reading the lists responsive the selected entries.

54. A method according to claim 53, wherein assigning the transport service instances to the queues comprises assigning the instances based on service parameters of the instances, and wherein reading the lists of descriptors comprises executing the descriptors responsive to the service parameters.

55. A method according to claim 50, wherein each of the descriptors occupies a given volume of space in the off-chip memory, and wherein writing the descriptors comprises generating outstanding read request descriptors, responsive to the incoming read request packets, up to a maximum number of incoming read request descriptors that can be outstanding at any given time as determined by the space available in the off-chip memory.

56. A method according to claim 50, wherein the off-chip memory to which the network interface adapter chip writes the descriptors is comprised in the system memory.

57. A method according to claim 50, wherein writing the descriptors comprises maintaining the plurality of the lists in a response database held in the off-chip memory for all the instances in common.

58. A method according to claim 57, wherein maintaining the plurality of the lists comprises assigning each of the instances a respective number of entries in the database to which its descriptors can be written.

59. A method according to claim 58, wherein the maintaining the plurality of the lists comprises

arranging the entries for each of the instances in the database as a cyclic buffer.